IN THE CLAIMS:

Please amend the claims as set forth in the following listing. This listing of the claims will replace all prior versions and listings of the claims in the present application.

1. (Currently amended) A method of modeling an arbitrarily complex environment, comprising:

providing a system comprising ~~defining, on~~ a model computer**[[,]]** for defining a ~~schema~~ data model having a plurality of types of data structures, wherein each type of data structure comprises one or more fields**[[;]]**, ~~storing, by~~ a database computer**[[**, the**]]** for storing a table schema in a static database, wherein the static database comprises a table for each type of data structure in the data model , wherein the plurality of types of data structures comprises components, relationships, properties and types**[[;]]**, and ~~representing, by~~ a database connectivity computer**[[,]]** for instantiating components from the plurality of types of data structures to represent entities in the arbitrarily complex environment;

instantiating a component from the plurality of types of data structures for representing each entity in the arbitrarily complex environment, wherein each instantiated component comprises a property field and check logic for determining the validity of a value in the property field; **[[**by**]]**

representing each instantiated component in a component table in the database;

assigning one or more values to the one or more fields in the database based on the attributes of the entity which the component is intended to represent;

instantiating relationships, wherein each instantiated relationship comprises a property field and check logic for determining the validity of the relationship;

representing each instantiated relationship in a relationship table in the database;

linking each relationship to at least two components; and

assigning one or more values based on the association which the relationship is intended to represent;

wherein ~~adding to or~~ altering the data model further comprises the steps of:

accessing, by a manager computer, the model computer, to add or alter a data structure, wherein the database computer is further configured to associate an added or altered

field with one or more tables, and wherein the database connectivity computer is further configured to associate a third value with the added property field or the altered property field; and

storing, by the model computer, the added or altered property field associated with the one or more data structures to the database, wherein the ~~addition or alteration of~~ the instantiation of components, relationships, properties and types associated with the added or altered fields does not change the defined data structures or the schema.

2. (Currently amended) The method of claim 1, wherein each component is instantiated . based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, and a set of properties and the schema comprises:

a propertyDefinition table, a componentType table, a component table, a ~~propertyTable~~ ~~category~~PropertyCategory table, a propertyDefinition table, a propertyValue table, a relationshipType table and a relationship table,

wherein ~~each~~ the ComponentType table comprises a set **[[**or**]]** of columns related to fields of a component type, wherein each component is linked to its type of component through a property field, whereby all instantiated components of a given type contain the same set of properties and check logic,

wherein the RelationshipType table comprises a set of columns related to fields of a relationship type, wherein each relationship is linked to its type of relationship through a property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyCategory table comprises a set of columns related to the definition of a property, wherein each property is linked to its type of relationship through a property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyDefinition table comprises a set of definitions of particular properties, wherein one or more properties contains a link to a field in

and a propertyValue table, wherein each property value comprises a link to a property definition in the propertyDefinition table and one of a link to a field in the ComponentType table or the RelationshipType table.

3. (Currently amended) The method of claim [[1]] 2, wherein each component type is in a hierarchy of component types, wherein a component subtype is a subgroup of a component type, wherein an instantiation of a component from a component subtype inherits all the property fields and check fields of the subtype and all the property fields and check fields of the parent component type.

4. (Currently amended) The method of claim 2, wherein a property field comprises a data type of one of a string, a numeric, a Boolean, a link, a date/time and a custom type, wherein each property field in a component is associated with a property of the entity being represented.

5. (Currently amended) The method of claim 2, wherein each property field comprises a data structure having a name, a description and a value.

6-9. (Cancelled).

10. (Currently amended) The method of claim 2, wherein each component is represented in the component table, wherein each component stored in the component table is linked to a type of component, wherein altering the model to include a new property field comprises the steps of:

adding a row to a table in the data schema, wherein the row comprises one or more fields containing the definition of the property, wherein the property definition is linked to a componentID field in the componentType table, whereby the added property field is associated with all components having the same component type.

11. (Previously Presented) The method of claim 10, wherein each component type is represented in the component type table, wherein each component type stored in the component type table is linked to a property in the property table.

12-15. (Cancelled).

16. (Currently amended) A system for modeling an arbitrarily complex environment, comprising:

a computer having a memory for storing a set of computer-executable instructions and a processor for executing the computer-executable instructions operable to:

define a schema having a ~~plurality~~ set of data structures, wherein each of the data structures comprises one or more fields;

instantiate components to represent physical and logical entities in the arbitrarily complex environment, wherein each instantiated component comprises a property field and check logic for determining the validity of a value in the property field;

[[by]] instantiate relationships to represent associations between two or more components in the arbitrarily complex environment, wherein each instantiated relationship comprises a property field and check logic for determining the validity of the relationship;

store the schema in a database; and

~~assigning~~ assign one or more values to the fields in the database; [[and]] wherein, when the data model is added to or altered, the system is further operable to:

store the added or altered data structure to the ~~database~~ schema, wherein the addition or alteration of components, relationships, properties and types do not change the defined data structures or the ~~schema~~ data model.

17. (Currently amended) The system of claim 16, wherein each component is instantiated based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, and a set of properties and the schema comprises:

a propertyDefinition table, a componentType table, a component table, a ~~propertyTable categoryPropertyCategory~~ table, a propertyDefinition table, a propertyValue table, a relationshipType table and a relationship table,

wherein ~~each~~ the ComponentType table comprises a set [[or]] of columns related to fields of a component type, wherein each component is linked to its type of component through a property field, whereby all instantiated components of a given type contain the same set of properties and check logic,

wherein the RelationshipType table comprises a set of columns related to fields of a relationship type, wherein each relationship is linked to its type of relationship through a property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyCategory table comprises a set of columns related to the definition of a property, wherein each property is linked to its type of relationship through a property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyDefinition table comprises a set of definitions of particular properties, wherein one or more properties contains a link to a field in

and a propertyValue table, wherein each property value comprises a link to a property definition in the propertyDefinition table and one of a link to a field in the ComponentType table or the RelationshipType table.

18. (Currently amended) The system of claim 17, wherein each component type is in a hierarchy of component types, wherein a component subtype is a subgroup of a component type, wherein an instantiation of a component from a component subtype inherits all the property fields and check fields of the subtype and all the property fields and check fields of the parent component type.

19. (Currently amended) The system of claim 18, wherein a property field comprises a data type of one of a string, a numeric, a Boolean, a link, a date/time and a custom type, wherein each property field in a component is associated with a property of the entity being represented.

20. (Currently amended) The system of claim 18, wherein a property field comprises a data structure having a name, a description and a value.

21-22. (Cancelled).

23. (Previously Presented) The system of claim 17, wherein each relationship type is a parent type or a subtype.

24-30. (Cancelled).

31. (Currently amended) A software product comprising a set of instructions stored on a computer-readable medium in a computer, wherein the computer has a computer memory and a processor for executing the set of instructions, wherein the software product comprises:

an instruction to define a schema having a ~~plurality~~ set of data structures, wherein each of the data structures comprises one or more fields; and

an instruction to instantiate components to represent physical and logical entities in the arbitrarily complex environment, wherein each instantiated component comprises a property field and check logic for determining the validity of a value in the property field;

[[by**]]** an instruction to instantiate relationships to represent associations between two or more components in the arbitrarily complex environment, wherein each instantiated relationship comprises a property field and check logic for determining the validity of the relationship;

an instruction to store the schema in a database; and

an instruction to ~~assigning~~ assign values to the fields in the database;

wherein, when the data model is added to or altered, the software product further comprises an instruction to

store the added or altered data structure to the ~~database~~ schema,

wherein the addition or alteration of components, relationships, properties and types do not change the defined data structures or the ~~schema~~ data model.

32. (Currently amended) The software product of claim 31, wherein each component is instantiated based on a generic component type and has a set of core attributes comprising an id, a name, a description, a type, and a set of properties and the schema comprises:

a propertyDefinition table, a componentType table, a component table, a ~~propertyTable categoryPropertyCategory~~ table, a propertyDefinition table, a propertyValue table, a relationshipType table and a relationship table,

wherein ~~each~~ the ComponentType table comprises a set **[[**or**]]** of columns related to fields of a component type, wherein each component is linked to its type of component through a property field, whereby all instantiated components of a given type contain the same set of properties and check logic,

wherein the RelationshipType table comprises a set of columns related to fields of a relationship type, wherein each relationship is linked to its type of relationship through a

property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyCategory table comprises a set of columns related to the definition of a property, wherein each property is linked to its type of relationship through a property field, whereby all instantiated relationships of a given type contain the same set of properties and check logic,

wherein the PropertyDefinition table comprises a set of definitions of particular properties, wherein one or more properties contains a link to a field in

and a propertyValue table, wherein each property value comprises a link to a property definition in the propertyDefinition table and one of a link to a field in the ComponentType table or the RelationshipType table.

33. (Currently amended) The software product of claim 32, wherein each component type is in a hierarchy of component types, wherein a component subtype is a subgroup of a component type, wherein an instantiation of a component from a component subtype inherits all the property fields and check fields of the subtype and all the property fields and check fields of the parent component type.

34. (Currently amended) The software product of claim 33, wherein a property field comprises a data type of one of a string, a numeric, a Boolean, a link, a date/time and a custom type, wherein each property field in a component is associated with a property of the entity being represented.

35. (Currently amended) The software product of claim 33, wherein a property field comprises a data structure having a name, a description and a value.

36-37. (Cancelled).

38. (Currently amended) The software product of claim **[[**37**]]** 35, wherein each relationship type is a parent type or a subtype.

39-53. (Cancelled).